

# Intégration continue

Tahiry Ratsimbahotra AUSY

# Enjeux

- Développement Parallèle (Gestion de conf)
- Reporting (qualité du produit, gestion de charges+estimation) mesure
- Test de no régression en temps réel/Minimiser l'impact des modifications: test
- Intégration de toute la chaîne de développement (spec => Livraison)
- AGILE prise en compte du cycle court
- Effort vs rentabilisation
- Livraison continue ne veut pas dire livraison à tout moment (décision humaine)

# Intérêts

- gestion de configuration (à chaque commit relance de la chaine complète)git,svn,clearcase
- Cycle en V ou AGILE
- Développement incrémental (composant développé , testeur prêt) multi language
- Automatisation des tests
- Automatisation de la compilation (makefile) (fabrication du binaire/build)
- Time to market (livraison continue/Correction rapide/patch/mise à jour )
- Utilisation de cluster ou grille de calcul
- Ne pas oublier une correction de bug quelque part
- Notification (mail, sms, twits)



# Mettre en place l'Intégration Continue au sein de votre organisation

- Phase 1 — build Manuel
  - Phase 2 — Builds automatisé
  - Phase 3 — Builds quotidiens et tests automatisés
  - Phase 4 — Métriques
  - Phase 5 — tests d'intégration
  - Phase 6 — Tests de recettes
  - Phase 7— Déploiement Continu
- 
- JENKINS ou HUDSON / MAVEN-ANT/ GIT / JAVA/JUNIT/JIRA/SONAR

# Métriques

- Nombre de lignes de code
- Pourcentage de code documenter
- Pourcentage de code dupliqué
- Problèmes dans le code ( bloquant, critique, majeur, mineur, informatif)
  - Possibilité de créer des tickets directement (si plugin installé)
- Tests unitaires
  - Pourcentage de couverture
  - Pourcentage des tests réussis

# JENKINS

- Commit des sources (git)
- Récupération des sources/Envoi des sources
- Lancement du Job Maven
- Demande des librairies (Nexus)
- Récupération des librairies dans le dépôt
- Envoi des librairies à Jenkins puis construction (Maven)
- Envoi des métriques (Sonar)
- Publications des librairies

# GITLAB CI

- Hébergement de code avec Gestion de versions
- Reporting (gestion de modification, bug reports, feedback with a fully featured Issue Tracker)
- Organisation et priorisation des demandes de support
- Relecture de Code et de requête de merge en ligne par branche
- Fabrication de binaire, test et déploiement
- Statistique
- Docker et GitLab Conteneurs
- Analyse du cycle de vie du logiciel (dev et utilisation)



# MOTS CLES

- Dockers : Virtualisation empaquetage d'un binaire et ses librairies de dépendances organisé sous forme de couches (empilements) dans un conteneurs
- Containers : localisation d'un paquet sous forme d'images
- Runners : lanceur de tâche sur une ou plusieurs VMs
- DevOps : Main dans la main le développeur et les utilisateurs pur limiter les conflits d'intérêt (qualité pour l'utilisateur , temps et coût pour le développeur)

# Gitlab

- Creation compte sur [gitlab.com](https://gitlab.com)
- Creation projet
- Cloner le projet en local
- Faire des modifications

# Pipeline (continuous integration)

- Créer fichier .gitlab-ci.yml
- # This file is a template, and might need editing before it works on your project.
- # before\_script:
- # - apt update && apt -y install make autoconf
- script:
  - - tenis ./st/test/stat.ptu
  - - gcc -Wall -O3 -g -gstabs /tmp/maintenis\_tmp.c ./st/src/math.c ./st/src/stat.c -I./st/src -o ./st/bin/prog -lm
  - artifacts:
    - paths: - ./st/bin/prog
    - # depending on your build setup it's most likely a good idea to cache outputs to reduce the build time # cache: # paths: # - "\*" # run tests using the binary built before
    - test: stage:
      - test script:
      - - export LC\_NUMERIC="C"
      - - echo "EXECUTION BINAIRE DE TEST DANS GDB"
      - - gdb --batch -x /tmp/user\_debug.gdb --args ./st/bin/prog
      - - tenispostpro test\_result.log

# Docker

- [Hub.docker.com](https://hub.docker.com)
- Layer organization
- Dockerfile

# Dockerfile

- FROM ubuntu
- MAINTAINER nom\_utilisateur <adressemail>
- RUN apt-get update
- RUN apt-get install outil
- ADD script /usr/bin/script
- CMD [« bash »]
- WORKDIR /usr/share/tmp
- VOLUME /volume/data
- ENV NOM\_VAR valeur

# Docker command

- Docker build -t katanougou/tenis\_c .
- Docker images
- Docker ps -a
- Docker run -it katanougou/tenis\_c
- Docker -d katanougou/tenis\_c (daemon)
- Docker rm -f id docker
- Docker commit id\_docker username/path
- Docker push username/path

- Introduction
- **Les outils intégrés (Maven/gradle, gitlab)**
- Gestion de conf
- Build
- Test
- Métrique
- **Mettre en place l'Intégration Continue au sein de votre organisation**
- **Optimases NWOW**